REMARKS

In view of the foregoing amendment and following remarks, Applicant respectfully requests favorable reconsideration of this application.

Applicant thanks the Office for the withdrawal of the objections to claims 1, 12, and 17-19 and the rejection under 35 U.S.C. §112, first paragraph, of claims 1-9 and 12-26 in view of Applicant's arguments and amendments.

In Section 8 of the Office Action, the Office objected to the use of trademarks in the specification without specific notice of the proprietary nature of the marks. Applicant has herein amended the specification as needed to capitalize or otherwise denote any proprietary marks not previously already designated appropriately.

In this final Office Action, the Office maintains the rejection under 35 U.S.C. §103(a) of all claims, namely, claims 1-9 and 12-26, as unpatentable over Hejlsberg in view of Ferstl. Since the rejections are a repeat of previous rejections, Applicant will herein focus on what appears to be the primary dispute between Applicant and the Office. Particularly, the Office is relying on Hejlsberg as disclosing most of the limitations of the claims, but relies on Ferstl for teaching the computational grid architecture in which a plurality of context derived models reside and which are selectively chosen for use in connection with the application description (e.g., OML document). It seems established that the Office recognizes that Hejlsberg does not teach the grid computing nature of the present claims. Furthermore, Applicant readily

concedes that Ferstl discloses a distributed computing technique in which a plurality of computers on a plurality of network nodes perform computing jobs.

Further, although not expressly addressed in any of the Office Actions, it is assumed that the Office does not contend that Ferstl contains any relevant disclosure with respect to automatically generating computer program code, as in the present invention. Rather, Ferstl merely contains a generic teaching of distributed computing.

The present invention teaches automatically developing object oriented programming (OOP) objects using a plurality of context derived models residing within a computational grid. A user generates a description of an application such as by using object meta language (OML). The application definition, e.g., the OML document, is submitted to a group of context derived models residing at various computational nodes on a grid. More particularly, a web service is used to parse the OML document and survey the available grid nodes in order to locate an available node (e.g., a style sheet or XML template) that is in accordance with the application parameters set forth in the OML document. In an exemplary embodiment, the grid nodes contain XSL style sheets or XML templates capable of generating completely coded applications from XML definitions. Alternately, the nodes may contain simple GIF generation modules or more complex applications such as JAVA applications, J2EE applications, or C/C++ applications. The web service selects the appropriate module based upon the OML definitions set forth by the user. Next, the OML document is provided to the selected node, which applies object description variables using a transform language to produce a defined output object. The defined output object is then returned to the user. The

invention eliminates the need for the programmer to generate the actual code for the desired object.

The proposed combination of Hejlsberg and Ferstl hardly suggests to the skilled artisan the present invention. One of the fundamental concepts of the present invention is the use of a computational grid with nodes containing a plurality of coding modules, such as XSL style sheets or XML templates, and permitting a software developer to create only a high level description of the code, such as an OML document, and then generating the code by finding the appropriate template in the grid and applying the high level description to that coding module to generate an output object.

In order for the claimed invention to be considered obvious in view of a combination of the teachings of two separate references, the combination of the teachings of those two references that would result in the present invention must be obvious to a person of ordinary skill in the related arts.

That is hardly the case here. As the Office recognizes, Hejlsberg does not teach a fundamental aspect of the present invention, namely the use of a computational grid with nodes containing different coding modules and selecting the most appropriate coding module to turn a high level description of the program into an actual program object. Ferstl discloses nothing but the generic concept of distributed computing. However, the present invention is much more than simply taking Hejlsberg and running it on a plurality of computers in a distributed computing environment. Thus, even if it would have been obvious to a person of ordinary skill in the related arts to combine

distributed computing, as taught by Ferstl, with Hejlsberg, it would teach the skilled artisan only the most fundamental step towards arriving at the present invention. If a generic disclosure of distributing computing such as Ferstl teaches anything relevant with respect to Hejlsberg, it is only that Hejlsberg's technique can be implemented using distributed computing. Thus, various sub-processes of Hejlsberg can be split among a plurality of computers operating at different nodes of a computer network. This hardly is equivalent to the presently claimed invention. There is no teaching in this proposed combination of the fundamental concept of providing a plurality of coding modules at different nodes of a computational grid and locating a suitable one of those nodes and then using it to generate an object.

Referring specifically to the claim language, independent claim 1 recites "providing a computational grid comprising a plurality of coding modules , , ,", "locating a suitable coding module in a node contained within the computational grid", and "applying said description to said suitable coding module to generate an output object".

Independent claim 12 recites "providing a computational grid comprising a plurality of coding modules . . .", "locating a suitable coding module in a node contained within the computational grid", "supplying said description to said node", and "applying said description to said suitable coding module to generate an output object".

Independent claim 19 recites "a computational grid, wherein said computational grid includes a plurality of computers sharing computational resources, said grid comprising a plurality of nodes, each node comprising at least one programming model" and "a web service for supplying said application description to a selected coding

module residing on said computational grid, wherein said coding module generates an object from said application description".

The prior art of record does not teach or reasonably suggest to an ordinarily skilled artisan the selection of a node out of many nodes that is suited to the particular code generation task at hand. The generic teachings of Ferstl, at best, would suggest merely partitioning a process among multiple computers, which is a very different concept than that of the present invention. In the present invention, the task is not apportioned. Rather, the task is performed by a suitable node.

Conclusion

The present invention is not taught or suggested by the prior art. Accordingly, the Examiner is respectfully requested to reconsider and withdraw the rejection of the claims. An early Notice of Allowance is earnestly solicited.

The Examiner is hereby authorized to charge any fees associated with this communication to Deposit Account No. 09-0461.

Respectfully submitted,

December 3, 2008
Date

/Theodore Naccarella/
Theodore Naccarella
Registration No. 33,023

SAUL EWING LLP
Centre Square West
1500 Market Street, 38th Floor
Philadelphia, PA 19102-2189
Telephone: 215 972 7877
Facsimile: 215 972 4161
Email: TNaccarella@saul.com